MULTIMEDIA UNIVERSITY

STUDENT ID NO

MULTIMEDIA UNIVERSITY

FINAL EXAMINATION

TRIMESTER 1, 2018/19

ECE2216 MICROCONTROLLER AND MICROPROCESSOR SYSTEMS

(All sections / Groups)

17 OCTOBER 2018
9:00 a.m. – 11:00 a.m.
(2 Hours)

INSTRUCTIONS TO STUDENT

1. This Question paper consists of 11 pages including cover page and appendix with 4 Questions only.

2. Attempt **ALL** questions. All questions carry equal marks and the distribution of the marks for each question is given.

3. Please print all your answers in the Answer Booklet provided.

Question 1

(a) Given hexadecimal $2B_{16}$ and octal 127_8 are both **signed** numbers in their binary:

i. Convert both of them into 8-bit binary format. [3 marks]

ii. Perform $2B_{16}$ **subtract** 127_8 arithmetic using their **binary numbers** obtained from Q1(a)(i). Show your answer in binary.
[2 marks]

iii. Identify the result obtained from Q1(a)(ii) as a positive or a negative number.
[1 mark]

iv. Find the decimal equivalent value of the result obtained from Q1(a)(ii). Show the computation steps. [2 marks]

Hint: Consider the numbers in Q1(a)(i) are in 2's complement representation.

(b) Sketch the **general structure** of a microcontroller. Give **three application examples** of microcontroller system.
[6 marks]

(c) A memory with 64MBytes of data capacity need to be accessed by a 16-bit microprocessor. Each memory location can store 16-bit data.

i. Identify the size of the data-bus required. [1 mark]

ii. Find the required size of the address-bus needed. Show the computation steps.
[3 marks]

iii. Find the maximum capacity (bits) that this address-bus able to access.
[2 marks]

(d) Describe **two characteristics** for Reduced Instruction Set Computer (RISC) and Complex Instruction Set Computer (CISC) respectively. State the type of **application** for RISC and CISC. [5 marks]

Continued ...

Question 2

(a) Name the 8051 control bus signals used to select external EPROMs and external RAMs.

[2 marks]

(b) After executing the instruction codes shown in Figure Q2(b), state the valus of the accumulator, carry flag, auxiliary carry flag and overflow flag.

```
MOV   R0,  #5FH
MOV    A,  #2CH
ADD    A,  R0
```

Figure Q2(b)

[6 marks]

(c) An 8051 microcontroller is required to be configured with 32KBytes of external ROM and 32KBytes of external RAM. Assume each RAM and ROM memory blocks are available as 32KBytes. Draw the configuration of the system showing the 8051 signal lines to be used for address, data and control buses.

[12 marks]

(d) Figure Q2(d) shows an 8051 microcontroller fetching an instruction (2BH) from the external memory location (10ABH). Explain the functions of ALE and PSEN.

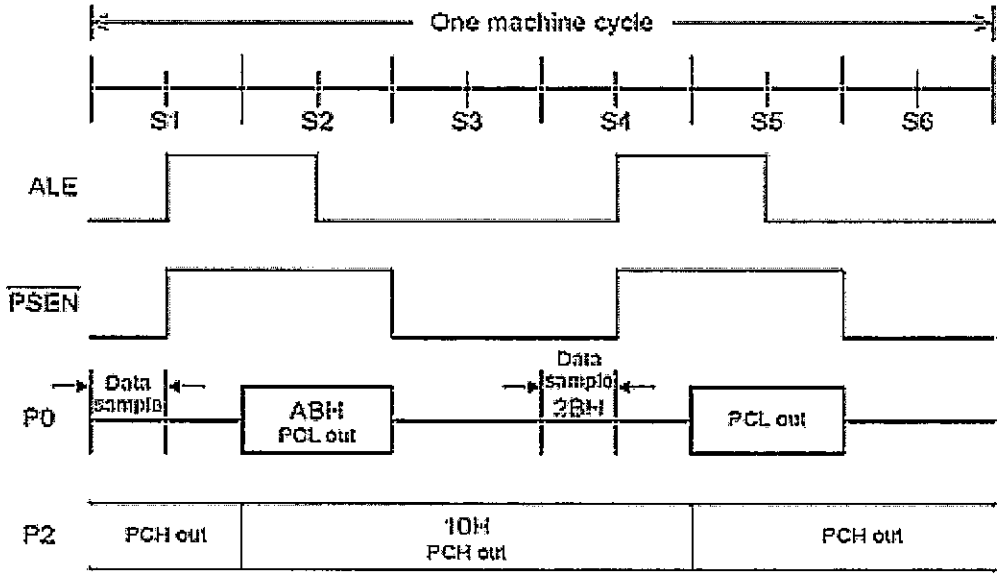


Figure Q2(d): Fetching an instruction (2BH) from the external memory location (10ABH).

[5 marks]

Continued...

Question 3

(a) The instruction sequence in Code Listing Q3(a) is written for the 8051 microcontroller.

```
MOV A, #5AH    ; first instruction
MOV R7, #0FFH  ; second instruction
XRL A, R7      ; third instruction
RR  A          ; forth instruction
```

Code Listing Q3(a)

i. Convert each instruction to the corresponding machine codes. [4 marks]

ii. Identify the addressing mode of the source and destination operands for the first and third instructions. [3 marks]

iii. Determine the size of this instruction sequence in bytes. [1 mark]

Hint: Refer to the opcode map in the Appendix.

(b) An embedded system engineer has developed an assembly program for the 8051 microcontroller as shown in Code Listing Q3(b).

```
      ORG 0000H
      MOV R0, #30H
      MOV R1, #1
      MOV R2, #1
      MOV R3, #4
LOOP: MOV A, R1
      ADD A, R2
      MOV @R0, A
      MOV B, R1
      MOV R2, B
      MOV R1, A
      INC R0
      DJNZ R3, LOOP
      SJMP $          ; program stops here
      END
```

Code Listing Q3(b)

Continued...

As a trainee in the company, you are required to perform the following tasks for the program listed in Code Listing Q3(b):

i. Analyze the program to determine the address of the affected memory locations in the internal RAM and their final contents after the program has been executed. [4 marks]

ii. Determine the final contents of registers R0 and R3 when the program has stopped. [2 marks]

iii. Calculate the address represented by the label "LOOP". [2 marks]

Hint: Refer to the opcode map in the Appendix.

(c) Write an 8051 instruction sequence to perform the following arithmetic operation and store the final result in register R1:

$$(93_{10} - 88_{10}) \times (18_{10} + 23_{10})$$

[4 marks]

(d) Develop an assembly program for the 8051 microcontroller to generate a 20 Hz square wave as shown in Figure Q3(d) continuously at pin P0.1 by using Timer 0 operating in Mode 1. Assume that the microcontroller is operating with a 12 MHz crystal oscillator. [5 marks]

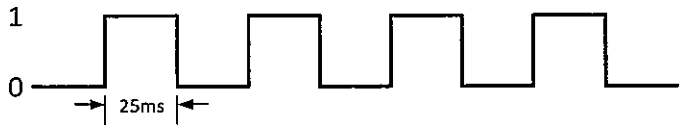


Figure Q3(d)

Hint: Refer to the opcode map in the Appendix.

Continued...

Question 4

(a) An 8051 microcontroller is equipped with an on-chip serial port for communication with external devices using UART.

i. Draw a block diagram to describe the hardware connection to make the RS232 compatible with the 8051 microcontroller. Briefly explain your block diagram. [2 marks]

ii. Design an instruction sequence to initialize the on-chip serial port to Mode 1 with baud rate of 19200 (with SMOD = 1) to transmit a character "H" serially and continuously. Use Timer 1 and Mode 2 with an oscillator frequency of 11.0592 MHz [5 marks]

iii. Write a subroutine called "Msg_sent" to transmit the string "Yes" serially via UART. [5 marks]

(b) As an intern in your company, you need to analyse an 8051 assembly code shown in Figure Q4(b). Determine the type of interrupt that have been enabled and hence the functionality of the code.

(Hints: The microcontroller is operating with an 11.0592MHz crystal oscillator. Assume active low connection for both LED.)

[8 marks]

(c) Sketch the circuit diagram to interface three common cathode seven-segment as digit0, digit1 and digit2 respectively to an 8051 microcontroller. Use P0 for seven segment display and P2 to select the digit via transistor 2N2222.

[5 marks]

Continued...

```
            ORG     0000H
            LJMP    main
            ORG     001BH
            LJMP    T1ISR

            ORG     0030H
main:       SETB    P2.2            ;Configured as input
            MOV     TMOD,#10H       ;configure timer1
            MOV     TH1,#04BH
            MOV     TL1,#0FDH       ;50ms
            MOV     IE.#088H
            SETB    TR1

loop:       SETB    P1.1            ;turn off LED2
            JB      P2.2,$
            CLR     P1.1            ;turn on LED2
            MOV     A,#20
            ACALL   Delay
            SJMP    loop

T1ISR:      CLR     TR1
            MOV     TH1,#4BH
            MOV     TL1,#0FDH
            CPL     P1.0            ;toggle LED1
            SETB    TR1
            RETI

Delay:      MOV     R0,A
Repeat:     MOV     R1,#10
Again:      MOV     R2,#10
Here:       NOP
            NOP
            DJNZ    R2,Here
            DJNZ    R1,Again
            DJNZ    R0,Repeat
            RET
```

Figure Q4(b)

End of Paper

APPENDIX

Special Function Register Formats

Interrupt Enable (IE)

Bit Addr.	AFH	-	-	ACH	ABH	AAH	A9H	A8H
Name	**EA**	**-**	**-**	**ES**	**ET1**	**EX1**	**ET0**	**EX0**

BIT	SYMBOL	FUNCTION (Enable=1, Disable=0)
IE.7	EA	Global enable/disable. EA = 1, each individual source is enabled/disabled by setting/clearing its enable bit. EA = 0, disable all interrupts.
IE.6	-	Undefined
IE.5	-	Not implemented in 8051. ET2 for 8052.
IE.4	ES	Serial port interrupt enable bit.
IE.3	ET1	Timer 1 interrupt enable bit.
IE.2	EX1	External interrupt enable bit.
IE.1	ET0	Timer 0 interrupt enable bit.
IE.0	EX0	External interrupt enable bit.

Interrupt Priority (IP)

Bit Addr.	-	-	-	BCH	BBH	BAH	B9H	B8H
Name	**-**	**-**	**-**	**PS**	**PT1**	**PX1**	**PT0**	**PX0**

BIT	SYMBOL	FUNCTION (Enable=1, Disable=0)
IP.7	-	Undefined
IP.6	-	Undefined
IP.5	-	Not implemented in 8051. PT2 for 8052.
IP.4	PS	Serial port interrupt priority bit.
IP.3	PT1	Timer 1 interrupt priority bit.
IP.2	PX1	External interrupt priority bit.
IP.1	PT0	Timer 0 interrupt priority bit.
IP.0	PX0	External interrupt priority bit.

Interrupt Vectors

Interrupt Source	Flag	Vector Address
System Reset	RST	0000H
External 0	IE0	0003H
Timer 0	TF0	000BH
External 1	IE1	0013H
Timer 1	TF1	001BH
Serial Port	RI & TI	0023H
Timer 2 (8052)	TF2 or EXF2	002BH

Program Status Word (PSW)

Bit Addr.	D7H	D6H	D5H	D4H	D3H	D2H	-	D0H
Name	**CY**	**AC**	**F0**	**RS1**	**RS0**	**0V**	-	**P**

Serial Control (SCON)

Bit Addr.	9FH	9EH	9DH	9CH	9BH	9AH	99H	98H
Name	**SM0**	**SM1**	**SM2**	**REN**	**TB8**	**RB8**	**TI**	**RI**

BIT	**SYMBOL**	**FUNCTION**
SCON.7	SM0	Serial port mode bit 0 (see Table A.1).
SCON.6	SM1	Serial port mode bit 1 (see Table A.1).
SCON.5	SM2	Serial port mode bit 2; enables multiprocessor communications in modes 2 and 3; RI will not be activated if received 9th bit is 0. In mode 1, if SM2 = 1, then RI will be activated only if a valid stop bit was received. In mode 0, SM2 should be 0.
SCON.4	REN	Receiver enable; must be set to receive characters.
SCON.3	TB8	Transmit bit 8; 9th bit transmitted in modes 2 and 3; set/cleared by software.
SCON.2	RB8	Receive bit 8; 9th bit received.
SCON.1	TI	Transmit interrupt flag; set at end of character transmission; cleared by software.
SCON.0	RI	Receive interrupt flag; set at end of character reception; cleared by software.

Table A.1 The 8051 Serial Port Mode Selection

SM0	**SM1**	**Mode**	**Description**	**Baud Rate**
0	0	0	Shift register	Fixed
0	1	1	8-bit UART	Variable
1	0	2	9-bit UART	Fixed
1	1	3	9-bit UART	Variable

PCON (Power Control)

Bit	7	6	5	4	3	2	1	0
Name	**SMOD**	-	-	-	**GF1**	**GF0**	**PD**	**IDL**

BIT	**SYMBOL**	**FUNCTION**
7	SMOD	Double the serial port baud rate when it is set to 1
3	GF1	General purpose flag bit 1
2	GF0	General purpose flag bit 0
1	PD	Activate the microcontroller power-down mode when it is set to 1
0	IDL	Activate the microcontroller idle mode when it is set to 1

Timer Control (TCON)

Bit Addr.	8FH	8EH	8DH	8CH	8BH	8AH	89H	88H
Name	**TF1**	**TR1**	**TF0**	**TR0**	**IE1**	**IT1**	**IE0**	**IT0**

BIT	SYMBOL	FUNCTION
TCON.7	TF1	Timer-1 overflow flag. Set by hardware on overflow. Cleared by hardware when processor vectors to interrupt routine. Must be cleared by software when not involve interrupt
TCON.6	TR1	Timer-1 run control bit. Set/cleared by software to turn timer/counter on/off.
TCON.5	TF0	Timer-0 overflow flag. Do the same function as TF1 but for Timer-0.
TCON.4	TR0	Timer-0 run control bit. Do the same function as TR1 but for Timer-0.
TCON.3	IE1	External interrupt-1 edge flag. Set by hardware when interrupt-1 falling edge is detected. Cleared by hardware when interrupt is processed.
TCON.2	IT1	Interrupt-1 Type control bit. Set/cleared by software to specify falling edge/low level triggered external interrupts.
TCON.1	IE0	External interrupt-1 edge flag. Do the same function as IE1 but for external interrupt-0.
TCON.0	IT0	Interrupt-0 Type control bit. Do the same function as IT1 but for external interrupt-0.

Timer Mode (TMOD)

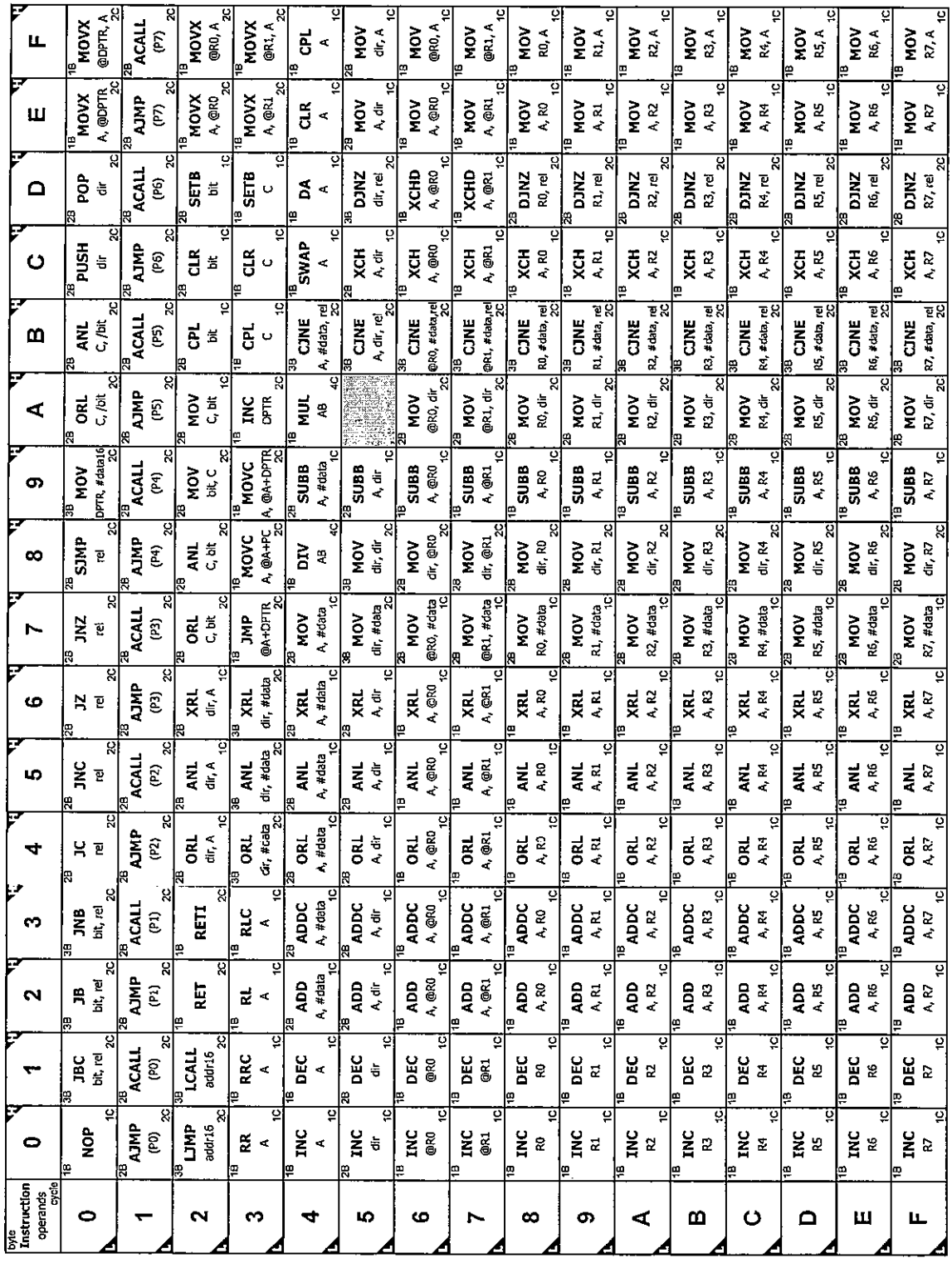
Bit	7	6	5	4	3	2	1	0
Name	**GATE**	**C/T**	**M1**	**M0**	**GATE**	**C/T**	**M1**	**M0**

BIT	SYMBOL	FUNCTION
TMOD.7	GATE1	When this bit is set the timer will only run when INT1 (P3.3) is high (hardware control). When this bit is cleared the timer will run regardless of the state of INT1 (software control).
TMOD.6	C/T1	Timer / Counter select bit. C / $\overline{T}$ = 0 → Timer operation. C / $\overline{T}$ = 1 → Counter operation.
TMOD.5	M1	Mode selection bits (see Table A.2). [for timer 1]
TMOD.4	M0	Mode selection bits (see Table A.2). [for timer 1]
TMOD.3	GATE0	Exactly the same function as GATE1 but for Timer0
TMOD.2	C/T0	Exactly the same function as C/T1 but for Timer0
TMOD.1	M1	Mode selection bits (see Table A.2). [for timer 0]
TMOD.0	M0	Mode selection bits (see Table A.2). [for timer 0]

Table A.2 Timer Mode Selection

M1	M0	Timer Mode	Description of Mode
0	0	0	13-bit Timer
0	1	1	16-bit Timer
1	0	2	8-bit auto-reload
1	1	3	Split timer mode

Opcode Map



byte / Instruction / operands / cycle	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	1B NOP 1C	3B JBC bit, rel 2C	3B JB bit, rel 2C	3B JNB bit, rel 2C	2B JC rel 2C	2B JNC rel 2C	2B JZ rel 2C	2B JNZ rel 2C	2B SJMP rel 2C	3B MOV DPTR, #data16 2C	2B ORL C, /bit 2C	2B ANL C, /bit 2C	2B PUSH dir 2C	2B POP dir 2C	1B MOVX A, @DPTR 2C	1B MOVX @DPTR, A 2C
1	2B AJMP (P0) 2C	2B ACALL (P0) 2C	2B AJMP (P1) 2C	2B ACALL (P1) 2C	2B AJMP (P2) 2C	2B ACALL (P2) 2C	2B AJMP (P3) 2C	2B ACALL (P3) 2C	2B AJMP (P4) 2C	2B ACALL (P4) 2C	2B AJMP (P5) 2C	2B ACALL (P5) 2C	2B AJMP (P6) 2C	2B ACALL (P6) 2C	2B AJMP (P7) 2C	2B ACALL (P7) 2C
2	3B LJMP addr16 2C	3B LCALL addr16 2C	1B RET 2C	1B RETI 2C	2B ORL dir, A 1C	2B ANL dir, A 1C	2B XRL dir, A 1C	2B ORL C, bit 2C	2B ANL C, bit 2C	2B MOV bit, C 2C	2B MOV C, bit 2C	2B CPL bit 1C	2B CLR bit 1C	2B SETB bit 1C	1B MOVX A, @R0 2C	1B MOVX @R0, A 2C
3	1B RR A 1C	1B RRC A 1C	1B RL A 1C	1B RLC A 1C	3B ORL dir, #data 2C	3B ANL dir, #data 2C	3B XRL dir, #data 2C	1B JMP @A+DPTR 2C	1B MOVC A, @A+PC 2C	1B MOVC A, @A+DPTR 2C	1B INC DPTR 2C	1B CPL C 1C	1B CLR C 1C	1B SETB C 1C	1B MOVX A, @R1 2C	1B MOVX @R1, A 2C
4	1B INC A 1C	1B DEC A 1C	2B ADD A, #data 1C	2B ADDC A, #data 1C	2B ORL A, #data 1C	2B ANL A, #data 1C	2B XRL A, #data 1C	2B MOV A, #data 1C	1B MOVC A, @A+DPTR 2C	2B SUBB A, #data 1C	1B MUL AB 4C	3B CJNE A, #data, rel 2C	1B SWAP A 1C	1B DA A 1C	1B CLR A 1C	1B CPL A 1C
5	2B INC dir 1C	2B DEC dir 1C	2B ADD A, dir 1C	2B ADDC A, dir 1C	2B ORL A, dir 1C	2B ANL A, dir 1C	2B XRL A, dir 1C	3B MOV dir, #data 2C	3B MOV dir, dir 2C	2B SUBB A, dir 1C	2B MOV A, dir 1C	3B CJNE A, dir, rel 2C	2B XCH A, dir 1C	3B DJNZ dir, rel 2C	2B MOV A, dir 1C	2B MOV dir, A 1C
6	1B INC @R0 1C	1B DEC @R0 1C	1B ADD A, @R0 1C	1B ADDC A, @R0 1C	1B ORL A, @R0 1C	1B ANL A, @R0 1C	1B XRL A, @R0 1C	2B MOV @R0, #data 1C	2B MOV dir, @R0 2C	1B SUBB A, @R0 1C	2B MOV @R0, dir 2C	3B CJNE @R0, #data,rel 2C	1B XCH A, @R0 1C	1B XCHD A, @R0 1C	1B MOV A, @R0 1C	1B MOV @R0, A 1C
7	1B INC @R1 1C	1B DEC @R1 1C	1B ADD A, @R1 1C	1B ADDC A, @R1 1C	1B ORL A, @R1 1C	1B ANL A, @R1 1C	1B XRL A, @R1 1C	2B MOV @R1, #data 1C	2B MOV dir, @R1 2C	1B SUBB A, @R1 1C	2B MOV @R1, dir 2C	3B CJNE @R1, #data,rel 2C	1B XCH A, @R1 1C	1B XCHD A, @R1 1C	1B MOV A, @R1 1C	1B MOV @R1, A 1C
8	1B INC R0 1C	1B DEC R0 1C	1B ADD A, R0 1C	1B ADDC A, R0 1C	1B ORL A, R0 1C	1B ANL A, R0 1C	1B XRL A, R0 1C	2B MOV R0, #data 1C	2B MOV dir, R0 2C	1B SUBB A, R0 1C	2B MOV R0, dir 2C	3B CJNE R0, #data, rel 2C	1B XCH A, R0 1C	2B DJNZ R0, rel 2C	1B MOV A, R0 1C	1B MOV R0, A 1C
9	1B INC R1 1C	1B DEC R1 1C	1B ADD A, R1 1C	1B ADDC A, R1 1C	1B ORL A, R1 1C	1B ANL A, R1 1C	1B XRL A, R1 1C	2B MOV R1, #data 1C	2B MOV dir, R1 2C	1B SUBB A, R1 1C	2B MOV R1, dir 2C	3B CJNE R1, #data, rel 2C	1B XCH A, R1 1C	2B DJNZ R1, rel 2C	1B MOV A, R1 1C	1B MOV R1, A 1C
A	1B INC R2 1C	1B DEC R2 1C	1B ADD A, R2 1C	1B ADDC A, R2 1C	1B ORL A, R2 1C	1B ANL A, R2 1C	1B XRL A, R2 1C	2B MOV R2, #data 1C	2B MOV dir, R2 2C	1B SUBB A, R2 1C	2B MOV R2, dir 2C	3B CJNE R2, #data, rel 2C	1B XCH A, R2 1C	2B DJNZ R2, rel 2C	1B MOV A, R2 1C	1B MOV R2, A 1C
B	1B INC R3 1C	1B DEC R3 1C	1B ADD A, R3 1C	1B ADDC A, R3 1C	1B ORL A, R3 1C	1B ANL A, R3 1C	1B XRL A, R3 1C	2B MOV R3, #data 1C	2B MOV dir, R3 2C	1B SUBB A, R3 1C	2B MOV R3, dir 2C	3B CJNE R3, #data, rel 2C	1B XCH A, R3 1C	2B DJNZ R3, rel 2C	1B MOV A, R3 1C	1B MOV R3, A 1C
C	1B INC R4 1C	1B DEC R4 1C	1B ADD A, R4 1C	1B ADDC A, R4 1C	1B ORL A, R4 1C	1B ANL A, R4 1C	1B XRL A, R4 1C	2B MOV R4, #data 1C	2B MOV dir, R4 2C	1B SUBB A, R4 1C	2B MOV R4, dir 2C	3B CJNE R4, #data, rel 2C	1B XCH A, R4 1C	2B DJNZ R4, rel 2C	1B MOV A, R4 1C	1B MOV R4, A 1C
D	1B INC R5 1C	1B DEC R5 1C	1B ADD A, R5 1C	1B ADDC A, R5 1C	1B ORL A, R5 1C	1B ANL A, R5 1C	1B XRL A, R5 1C	2B MOV R5, #data 1C	2B MOV dir, R5 2C	1B SUBB A, R5 1C	2B MOV R5, dir 2C	3B CJNE R5, #data, rel 2C	1B XCH A, R5 1C	2B DJNZ R5, rel 2C	1B MOV A, R5 1C	1B MOV R5, A 1C
E	1B INC R6 1C	1B DEC R6 1C	1B ADD A, R6 1C	1B ADDC A, R6 1C	1B ORL A, R6 1C	1B ANL A, R6 1C	1B XRL A, R6 1C	2B MOV R6, #data 1C	2B MOV dir, R6 2C	1B SUBB A, R6 1C	2B MOV R6, dir 2C	3B CJNE R6, #data, rel 2C	1B XCH A, R6 1C	2B DJNZ R6, rel 2C	1B MOV A, R6 1C	1B MOV R6, A 1C
F	1B INC R7 1C	1B DEC R7 1C	1B ADD A, R7 1C	1B ADDC A, R7 1C	1B ORL A, R7 1C	1B ANL A, R7 1C	1B XRL A, R7 1C	2B MOV R7, #data 1C	2B MOV dir, R7 2C	1B SUBB A, R7 1C	2B MOV R7, dir 2C	3B CJNE R7, #data, rel 2C	1B XCH A, R7 1C	2B DJNZ R7, rel 2C	1B MOV A, R7 1C	1B MOV R7, A 1C